OmniMVS: End-to-End Learning for Omnidirectional Stereo Matching

Changhee Won, Jongbin Ryu and Jongwoo Lim*
Department of Computer Science, Hanyang University, Seoul, Korea.
{chwon, jongbinryu, jlim}@hanyang.ac.kr

Abstract

In this paper, we propose a novel end-to-end deep neural network model for omnidirectional depth estimation from a wide-baseline multi-view stereo setup. The images captured with ultra wide field-of-view (FOV) cameras on an omnidirectional rig are processed by the feature extraction module, and then the deep feature maps are warped onto the concentric spheres swept through all candidate depths using the calibrated camera parameters. The 3D encoder-decoder block takes the aligned feature volume to produce the omnidirectional depth estimate with regularization on uncertain regions utilizing the global context information. In addition, we present large-scale synthetic datasets for training and testing omnidirectional multi-view stereo algorithms. Our datasets consist of 11K ground-truth depth maps and 45K fisheye images in four orthogonal directions with various objects and environments. Experimental results show that the proposed method generates excellent results in both synthetic and real-world environments, and it outperforms the prior art and the omnidirectional versions of the state-of-the-art conventional stereo algorithms.

1. Introduction

Image-based depth estimation, including stereo and multi-view dense reconstruction, has been widely studied in the computer vision community for decades. In conventional two-view stereo matching, deep learning methods [12, 4] have achieved drastic performance improvement recently. Besides, there are strong needs on omnidirectional or wide FOV depth sensing in autonomous driving and robot navigation to sense the obstacles and surrounding structures. Human drivers watch all directions, not just the front, and holonomic robots need to sense all directions to move freely. However, conventional stereo rigs and algorithms cannot capture or estimate ultra wide FOV (>180°) depth maps. Merging depth maps from multiple conventional stereo pairs can be one possibility, but the useful global context information cannot be propagated between the pairs and there might be a discontinuity at the seam.

Recently, several works have been proposed for the omnidirectional stereo using multiple cameras [29], reflective mirrors [25], or wide FOV fisheye lenses [6]. Nevertheless, very few works utilize deep neural networks for the omnidirectional stereo. In SweepNet [30] a convolutional neural network (CNN) is used to compute the matching costs of equirectangular image pairs warped from the ultra-wide FOV images. The result cost volume is then refined by cost aggregation (*e.g.*, Semi-global matching [10]), which is a commonly used approach in conventional stereo matching [5, 32, 15]. However, such an approach may not be optimal in the wide-baseline omnidirectional setup since the occlusions are more frequent and heavier, and there can be multiple true matches for one ray (Fig. 2b). On the other hand, recent methods for conventional stereo matching such as GC-Net [14] and PSMNet [4] employ the end-to-end deep learning without separate cost aggregation, and achieve better performance compared to the traditional pipeline [32, 8, 26].

We introduce a novel end-to-end deep neural network for estimating omnidirectional depth from multi-view fisheye images. It consists of three blocks, unary feature extraction, spherical sweeping, and cost volume computation as illustrated in Fig. 1. The deep features built from the input images are warped to spherical feature maps for all hypothesized depths (spherical sweeping). Then a 4D feature volume is formed by concatenating the spherical feature maps from all views so that the correlation between multiple views can be learned efficiently. Finally, the 3D encoder-decoder block computes a regularized cost volume in consideration of the global context for omnidirectional depth estimation. While the proposed algorithm can handle various camera layouts, we choose the rig in Fig. 2a because it provides good coverage while it can be easily adopted in the existing vehicles.

Large-scale data with sufficient quantity, quality, and diversity are essential to train robust deep neural networks. Nonetheless, acquiring highly accurate dense depth measurements in real-world is very difficult due to the limitations of available depth sensors. Recent works [16, 21] have

*Corresponding author.

proposed to use realistically rendered synthetic images with ground truth depth maps for conventional stereo methods. Cityscape synthetic datasets in [30] are the only available datasets for the omnidirectional multi-view setup, but the number of data is not enough to train a large network, and they are limited to the outdoor driving scenes with few objects. In this work, we present complementary large-scale synthetic datasets in both indoor and outdoor environments with various objects.

The contributions of this paper are summarized as:

(i) We propose a novel end-to-end deep learning model to estimate an omnidirectional depth from multiple fisheye cameras. The proposed model directly projects feature maps to the predefined global spheres, combined with the 3D encoder-decoder block enabling to utilize global contexts for computing and regularizing the matching cost.

(ii) We offer large-scale synthetic datasets for the omnidirectional depth estimation. The datasets consist of multiple input fisheye images with corresponding omnidirectional depth maps. The experiments on the real-world environments show that our datasets successfully train our network.

(iii) We experimentally show that the proposed method outperforms the previous multi-stage methods. We also show that our approaches perform favorably compared to the omnidirectional versions of the state-of-the-art conventional stereo methods through extensive experiments.

2. Related Work

Deep Learning-based Methods for Conventional Stereo Conventional stereo setup assumes a rectified image pair as the input. Most traditional stereo algorithms before deep learning follow two steps: matching cost computation and cost aggregation. As summarized in Hirschmuller *et al*. [11], sum of absolute differences, filter-based cost, mutual information, or normalized cross-correlation are used to compute the matching cost, and for cost aggregation, local correlation-based methods, global graph cuts [2], and semi-global matching (SGM) [10] are used. Among them, SGM [10] is widely used because of its high accuracy and low computational overhead.

Recently, deep learning approaches report much improved performance in the stereo matching. Zagoruyko *et al*. [31] propose a CNN-based similarity measurement for image patch pairs. Similarly, Zbontar and LeCun [32] introduce MC-CNN that computes matching costs from small image patch pairs. Meanwhile, several papers focus on the cost aggregation or disparity refinement. Güney and Geiger [8] introduce Displets resolving matching ambiguities on reflection or textureless surfaces using objects' 3D models. Seki and Pollefeys [26] propose SGM-Net which predicts the smoothness penalties in SGM [10].

On the other hand, there have been several works on end-to-end modeling of the stereo pipeline. Kendall *et al*. [14] propose GC-Net which regularizes the matching cost by 3D convolutional encoder-decoder architecture, and performs disparity regression by the softargmin. Further, PSMNet by Chang and Chen [4] consists of spatial pyramid pooling modules for larger receptive field and multiply stacked 3D encoder-decoder architecture for learning more context information. Also, Mayer *et al*. [16] develop DispNet, an end-to-end network using correlation layers for disparity estimation, and it is further extended by Pang *et al*. [18] (CRL) and Ilg *et al*. [12] (DispNet-CSS). These end-to-end networks have achieved better performance compared to the conventional multi-stage methods.

Synthetic Datasets for Learning Stereo Matching For successful training of deep neural networks, an adequate large-scale dataset is essential. In stereo depth estimation, Middlebury [24, 11, 23] and KITTI datasets [7, 17] are most widely used. These databases are faithfully reflecting the real world, but capturing the ground truth depth requires complex calibration and has limited coverage, and more importantly, the number of images is often insufficient for training large networks.

Nowadays synthetically rendered datasets are used to complement the real datasets. Mayer *et al*. [16] introduce a large scale dataset for disparity, optical flow, and scene flow estimation. The proposed dataset consists of 2K scene images and dense disparity maps generated via rendering, which is $10\times$ larger than KITTI [17]. Ritcher *et al*. [20] provide fully annotated training data by simulating a living city in a realistic 3D game world. For semantic scene completion, SUNGC dataset [27] contains 45K synthetic indoor scenes of 400K rooms and 5M objects with depth and voxel maps. However, almost all datasets use single or stereo pinhole camera models with limited FOV, and there are very few datasets for omnidirectional stereo.

Omnidirectional Depth Estimation Various algorithms and systems have been proposed for the omnidirectional depth estimation [6, 25, 29], but very few use deep neural networks. Schönbein *et al*. [25] use two horizontally mounted 360°-FOV catadioptric cameras, and estimate the disparity from rectified omnidirectional images. Using two vertically mounted ultra-wide FOV fisheye cameras, Gao and Shen [6] estimate omnidirectional depth by projecting the input images onto four predefined planes. Im *et al*. [13] propose a temporal stereo algorithm that estimates an all around depth of the static scene from a short motion clip. Meanwhile, purely learning-based approaches Zioulis *et al*. [33] and Payen *et al*. [19] have been proposed estimating a 360° depth from a single panoramic image.

Recently, Won *et al*. [30] propose SweepNet with a

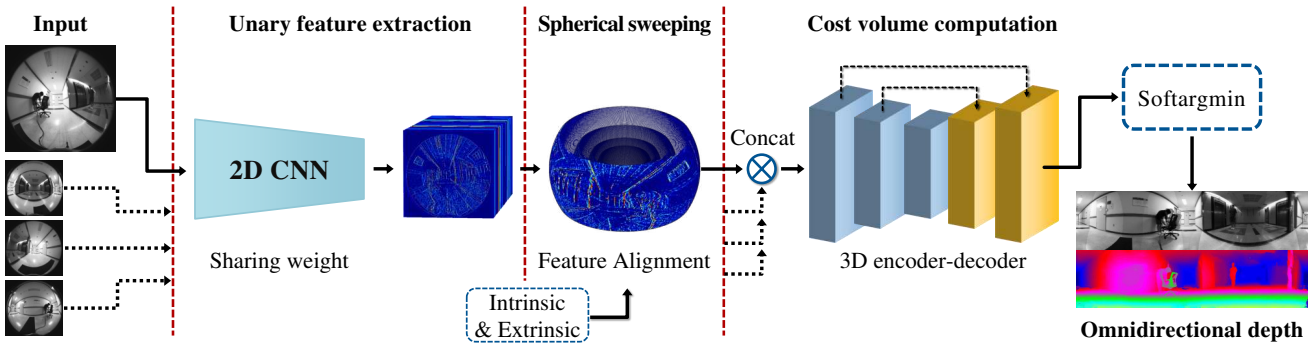


Figure 1: **Overview of the proposed method.** Each input image is fed into the 2D CNN for extracting feature maps. We project the unary feature maps into spherical features to build the matching cost volume. The final depth is acquired through cost volume computation by the 3D encoder-decoder architecture and softargmin.

multi-camera rig system for the omnidirectional stereo. They warp the input fisheye images onto the concentric global spheres, and SweepNet computes matching costs from the warped spherical images pair. Then, the cost volume is refined by applying SGM [10]. However, SGM cannot handle the multiple true matches occurring in such global sweeping approaches as in Fig. 2b.

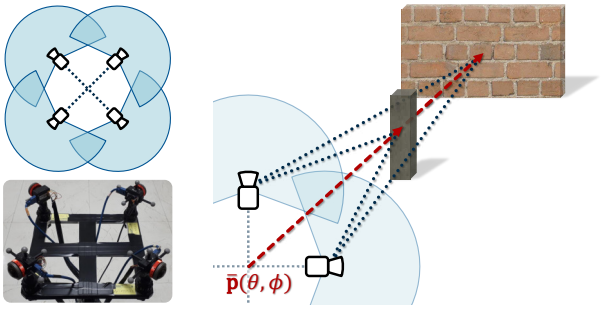
In this paper, we present the first end-to-end deep neural network for the omnidirectional stereo and large-scale datasets to train the network. As shown in the experiments, the proposed method achieves better performance compared to the previous methods and performs favorably in the real-world environment with our new datasets.

3. Omnidirectional Multi-view Stereo

In this section, we introduce the multi-fisheye camera rig and the spherical sweeping method, and then describe the proposed end-to-end network architecture for the omnidirectional stereo depth estimation. As shown in Fig. 1 our algorithm has three stages, unary feature extraction, spherical sweeping, and cost volume computation. In the following subsections, the individual stages are described in detail.

3.1. Spherical sweeping

The rig consists of multiple fisheye cameras mounted at fixed locations. Unlike the conventional stereo which uses the reference camera's coordinate system, we use the rig coordinate system for depth representation, as in [30]. For convenience we set the y-axis to be perpendicular to the plane closest to all camera centers, and the origin at the center of the projected camera centers. A unit ray $\bar{\mathbf{p}}$ for the spherical coordinate $\langle\theta, \phi\rangle$ corresponds to $\bar{\mathbf{p}}(\theta, \phi) = (\cos(\phi)\cos(\theta), \sin(\phi), \cos(\phi)\sin(\theta))^\top$. With the intrinsic and extrinsic parameters of the i-th camera (calibrated using [22, 28, 1]), the image pixel coordinate $\mathbf{x}_i$ for a 3D point $\mathbf{X}$ can be written as a projection function Π_i; $\mathbf{x}_i = \Pi_i(\mathbf{X})$. Thus a point at $\langle\theta, \phi\rangle$ on the sphere of radius ρ is projected to $\Pi_i(\rho\,\bar{\mathbf{p}}(\theta, \phi))$ in the i-th fisheye image.



(a) (b)

Figure 2: (a) **Wide-baseline multi-camera rig system.** (b) **Multiple true matches problem.** There can be several observations on a ray in such global sweeping approach.

Spherical sweeping generates a series of spheres with different radii and builds the spherical images of each input image. Similar to plane sweeping in conventional stereo, the inverse radius d_n is swept from 0 to d_{max}, where $1/d_{max}$ is the minimum depth to be considered and N is the number of spheres. The pixel value of the equirectangular spherical image warped onto n-th sphere is determined as

$$S_{n,i}(\theta, \phi) = I_i(\Pi_i(\bar{\mathbf{p}}(\theta, \phi)/d_n)), \tag{1}$$

where I_i is the input fisheye image captured by i-th camera and d_n is the n-th inverse depth.

3.2. Feature Learning and Alignment

Instead of using pixel intensities, recent stereo algorithms use deep features for computing matching costs. MC-CNN [32] shifts the right features by $-k$ pixels to align them with the left features, so as to compute the cost for k disparity by 1×1 convolutional filters. Further, GC-Net [14] builds a 4D cost volume by shifting and concatenating the feature maps across each disparity, so that it can be regular-

ized by a 3D CNN. In this way, the network can utilize geometric context (e.g., for handling occlusion) by depth-wise convolution, and also, the simple shifting operation makes gradient back-propagation easy. However, these approaches are limited to rectified conventional stereo, and cannot be applied to multi-view images in wide FOV or omnidirectional setups.

Instead of extracting features from the spherical images at all spheres, we choose to build a feature map in the input fisheye image space and warp the feature map according to Eq. 1. This saves huge amount of computation, and the impact on performance is minimal since the distortion in the original image is learned by the feature extraction network. The unary feature map $U = F_{CNN}(I)$ has $\frac{1}{r}H_I \times \frac{1}{r}W_I \times C$ resolution, where F_{CNN} is a 2D CNN for the feature extraction, H_I and W_I are the height and width of the input image, r is the reduction factor, and C is the number of channels.

The unary feature maps of the input images are then projected onto the predefined spheres. Following Eq. 1, the spherical feature map at n-th sphere for i-th image is determined as

$$\mathcal{S}_i(\phi, \theta, n, c) = U_c\left(\frac{1}{r}\Pi_i(\bar{\mathbf{p}}(\theta, \phi)/d_n)\right), \qquad (2)$$

where θ varies from $-\pi$ to π, and ϕ varies up to $\pm\pi/2$ according to the resolution. To ensure sufficient disparities between neighboring warped feature maps and to reduce the memory and computation overhead, we use every other spheres, i.e., $n \in [0, 2, \ldots, N-1]$, to make the warped 4D feature volume $\mathcal{S}_i$ of the size $H \times W \times \frac{N}{2} \times C$. With the calibrated intrinsic and extrinsic parameters, we use the coordinate lookup table and 2D bilinear interpolation in warping the feature maps, and during back-propagation the gradients are distributed inversely. We compute the validity mask M_i for each input image, and the pixels outside the valid region are ignored both in warping and back-propagation.

Finally, all spherical feature volumes $\{\mathcal{S}_i\}$ are merged and used as the input of the cost computation network. Our approaches enables the network to learn finding omnidirectional stereo correspondences from multiple fisheye images, and to utilize spherical geometric context information for the regularization by applying a 3D CNN to the spherical features.

3.3. Network Architecture

The architecture of the proposed network is detailed in Table 1. The input of the network is a set of grayscale fisheye images. We use the residual blocks [9] for the unary feature extraction, and the dilated convolution for the larger receptive field. The output feature map size is half ($r = 2$) of the input image. Each feature map is aligned by the spherical sweeping (Sec. 3.2), and transferred to the spherical feature by a 3×3 convolution. The spherical feature maps are concatenated and fused into the 4D initial cost volume by a $3 \times 3 \times 3$ convolution. We then use the 3D encoder-decoder architecture [14] to refine and regularize the cost volume using the global context information.

	Name	Layer Property	Output (H, W, N, C)
Unary feature extraction	Input		$H_I \times W_I$
	conv1	$5 \times 5, 32$	$1/2H_I \times 1/2W_I \times 32$
	conv2	$3 \times 3, 32$	
	conv3	$3 \times 3, 32$, add conv1	
	conv4-11	repeat conv2-3	
	conv12-17	repeat conv2-3 with dilate = 2, 3, 4	
Spherical sweeping	warp		$H \times W \times 1/2N \times 32$
	transference	$3 \times 3 \times 1, 32$	$1/2 \times 1/2 \times 1/2 \times 32$
	concat(4)*		$1/2 \times 1/2 \times 1/2 \times 128$
	fusion	$3 \times 3 \times 3, 64$	$1/2 \times 1/2 \times 1/2 \times 64$
Cost volume computation	3Dconv1-3	$3 \times 3 \times 3, 64$	$1/2 \times 1/2 \times 1/2 \times 64$
	3Dconv4-6	from 1, $3 \times 3 \times 3, 128$	$1/4 \times 1/4 \times 1/4 \times 128$
	3Dconv7-9	from 4, $3 \times 3 \times 3, 128$	$1/8 \times 1/8 \times 1/8 \times 128$
	3Dconv10-12	from 7, $3 \times 3 \times 3, 128$	$1/16 \times 1/16 \times 1/16 \times 128$
	3Dconv13-15	from 10, $3 \times 3 \times 3, 256$	$1/32 \times 1/32 \times 1/32 \times 256$
	3Ddeconv1	$3 \times 3 \times 3, 128$, add 3Dconv12	$1/16 \times 1/16 \times 1/16 \times 128$
	3Ddeconv2	$3 \times 3 \times 3, 128$, add 3Dconv9	$1/8 \times 1/8 \times 1/8 \times 128$
	3Ddeconv3	$3 \times 3 \times 3, 128$, add 3Dconv6	$1/4 \times 1/4 \times 1/4 \times 128$
	3Ddeconv4	$3 \times 3 \times 3, 64$, add 3Dconv3	$1/2 \times 1/2 \times 1/2 \times 64$
	3Ddeconv5	$3 \times 3 \times 3, 1$	$H \times W \times N$
	softargmin		$H \times W$

Table 1: The input images pass separately from conv1 to transference, then are merged by concat and fusion. H, W, and N are omitted for brevity. In this work we use 4 cameras, thus concat outputs $32 \times 4 = 128$ channels.

Finally, the inverse depth index $\hat{n}$ can be computed by the softargmin [14] as

$$\hat{n}(\theta, \phi) = \sum_{n=0}^{N-1} n \times \frac{e^{-\mathcal{C}(\phi,\theta,n)}}{\sum_\nu e^{-\mathcal{C}(\phi,\theta,\nu)}}$$

where $\mathcal{C}$ is the $(H \times W \times N)$ regularized cost volume.

To train the network in an end-to-end fashion, we use the input images and the ground truth inverse depth index as

$$n^*(\theta, \phi) = (N-1)\frac{d^*(\theta, \phi) - d_0}{d_{N-1} - d_0},$$

where $d^*(\cdot) = 1/\mathcal{D}^*(\cdot)$ is the ground truth inverse depth, and d_0 and d_{N-1} are the min and max inverse depth respectively. We use the absolute error loss between the ground truth and predicted index as

$$L(\theta, \phi) = \frac{1}{\sum_i M_i(\theta, \phi)}\left|\hat{n}(\theta, \phi) - \text{round}(n^*(\theta, \phi))\right|.$$

We use the stochastic gradient descent with a momentum to minimize the loss. The overall flow of the proposed network is illustrated in Fig. 1.

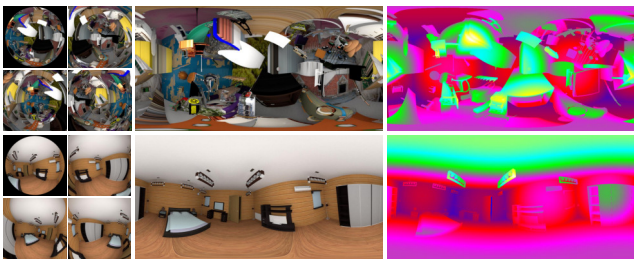


Figure 3: **Examples of our proposed datasets.** From left: input fisheye images with visibility (left-top), reference panorama image, and ground truth inverse depth map.

Dataset		# Training Scenes	# Training Frames	# Test Frames
SceneFlow [16]	FlyingThings3D	2247	21818	4248
	Monkaa	8	8591	-
	Driving	1	4392	-
Won *et al.* [30]	Sunny	1	700	300
	Cloudy	1	700	300
	Sunset	1	700	300
Ours	OmniThings	9216	9216	1024
	OmniHouse	451	2048	512

Table 2: **Comparison with the published datasets.** Our datasets have more training scenes and the comparable number of training and test frames to exsiting datasets.

4. Datasets

Although there exist many datasets for conventional stereo [7, 17, 16], only one dataset [30] is available for the omnidirectional stereo, but it only contains the outdoor road scenes. Therefore we create new synthetic datasets for more generic scenes and objects. Our datasets contain input fisheye images, omnidirectional depth maps, and reference panorama images. In addition to [30], we generate two much larger datasets (OmniThings and OmniHouse) in different environments using Blender.

OmniThings Similar to [16], OmniThings dataset consists of randomly generated objects around the camera rig. We collect 33474 3D object models from ShapeNet [3] and 4711 textures from Flickr and ImageAfter[1]. For each scene, we randomly choose 64 objects and place them onto the N spheres with random positions, rotations, scales, and textures, so that complex shapes of various objects and occlusions can be learned. We also place a randomly shaped room or sky for learning the background depth. The dataset has 9216 scenes for training and 1024 scenes for test.

OmniHouse In order to generate realistic indoor scenes, we reproduce the SUNCG dataset [27] which consists of 45K synthetic indoor scenes. We collect 451 house models from the SUNCG dataset, and place the virtual camera rig in them with random positions and orientations. We render 2048 frames for training and 512 frames for test.

[1]https://www.flickr.com and http://www.imageafter.com

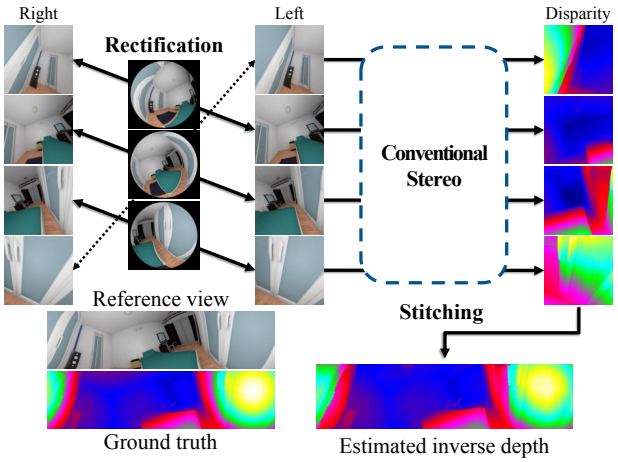


Figure 4: We rectify the input images into 512×512 and $120°$ FOV left-right pairs. The predicted disparity maps are merged into a $H \times W$ omnidirectional inverse depth index.

The overview of our proposed datasets is described in Table 2, and the examples are shown in Fig. 3. Each frame consists of four $220°$ FOV fisheye images, which have a resolution of $H_I = 768$ and $W_I = 800$, and one ground truth omnidirectional depth map, which has $H = 360$ and $W = 640$ (θ ranges from $-\pi$ to π and ϕ from $-\pi/2$ to $\pi/2$). In the next section, we show that the networks trained with our datasets successfully estimate the omnidirectional depth in the real-world environments, which proves the effectiveness of our synthetic datasets.

5. Experimental Results

5.1. Implementation and Training Details

To train the network, the input images are converted to grayscale, and the validity mask is set to only contain the pixels within $220°$ FOV. The intensity values in the valid area are then normalized to zero-mean and unit variance. To prevent the encoder-decoder network from learning only the valid regions in each channel, the order of feature maps to the concatenating stage is randomly permuted (*e.g.*, 1-2-3-4, 2-3-4-1, 3-4-1-2, or 4-1-2-3). Further, we randomly rotate the rig coordinate system (and the GT depth map accordingly) with a small angle, so that the network is not tightly coupled to specific layouts. In all our experiments, the output and GT depth maps are cropped to $H = 160$ ($-\pi/4 \leq \phi \leq \pi/4$) and $W = 640$ since the regions near the poles are highly distorted and less useful. The number of sweep spheres is set to $N = 192$. We train our network for 30 epochs on the OmniThings dataset from scratch, using 4096 training scenes. The learning rate λ is set to 0.003 for the first 20 epochs and 0.0003 for the remaining 10 epochs. We also test the network fine-tuned on the Sunny and Omni-

Dataset	OmniThings					OmniHouse				
Metric	>1	>3	>5	MAE	RMS	>1	>3	>5	MAE	RMS
Spherical sweeping with regularization										
ZNCC+SGM [10]	72.56	54.01	45.63	10.51	16.44	44.05	20.64	13.57	3.08	7.05
MC-CNN [32]+SGM	67.19	47.43	39.49	8.65	13.66	38.01	15.86	9.46	2.08	4.15
SweepNet [30]+SGM	67.20	47.63	39.66	8.87	13.90	36.60	15.41	9.36	2.07	4.38
Stitching conventional stereo										
PSMNet [4]	86.25	63.23	44.84	7.28	11.15	63.22	26.43	15.39	5.82	13.88
PSMNet-ft	82.69	51.98	41.74	9.09	13.71	87.56	27.01	12.89	3.51	6.05
DispNet-CSS [12]	50.62	27.77	19.50	4.06	7.98	*26.56	11.69	7.16	*1.54	*3.18
DispNet-CSS-ft	67.86	48.08	38.57	7.81	12.27	36.47	14.98	8.29	1.81	3.44
OmniMVS	**47.72**	**15.12**	**8.91**	**2.40**	**5.27**	30.53	*10.29	*6.27	1.72	4.05
OmniMVS-ft	*50.28	*22.78	*15.60	*3.52	*7.44	**21.09**	**4.63**	**2.58**	**1.04**	**1.97**

Dataset	Sunny					Cloudy					Sunset				
Metric	>1	>3	>5	MAE	RMS	>1	>3	>5	MAE	RMS	>1	>3	>5	MAE	RMS
Spherical sweeping with regularization															
ZNCC+SGM	52.00	21.45	10.96	2.50	5.35	53.09	22.17	11.50	2.58	5.45	52.33	21.90	11.29	2.53	5.31
MC-CNN+SGM	39.42	11.73	6.08	1.83	4.56	43.16	11.95	5.82	1.85	4.46	39.67	12.82	6.28	1.86	4.59
SweepNet+SGM	24.82	6.91	4.28	1.31	3.79	34.97	9.51	5.09	1.55	3.96	24.92	7.25	4.46	1.32	3.80
Stitching conventional stereo															
PSMNet	65.09	30.87	13.13	2.54	4.03	63.62	28.51	10.40	2.45	4.26	63.83	28.41	10.00	2.43	4.11
PSMNet-ft	92.67	31.45	21.32	4.33	7.76	92.92	31.24	20.14	4.13	7.32	93.24	30.64	19.65	4.11	7.43
DispNet-CSS	*24.80	8.54	5.59	1.44	4.02	*25.16	8.47	5.50	1.43	3.92	*24.79	8.29	5.34	1.38	3.76
DispNet-CSS-ft	39.02	21.12	14.47	2.37	4.85	42.29	21.55	14.28	2.43	4.88	40.21	20.91	14.43	2.40	4.88
OmniMVS	27.16	*6.13	*3.98	*1.24	*3.09	28.13	*5.37	*3.54	*1.17	*2.83	26.70	*6.19	*4.02	*1.24	*3.06
OmniMVS-ft	**13.93**	**2.87**	**1.71**	**0.79**	**2.12**	**12.20**	**2.48**	**1.46**	**0.72**	**1.85**	**14.14**	**2.88**	**1.71**	**0.79**	**2.04**

Table 3: **Quantitative comparison with other methods.** The error is defined in Eq. 3. The qualifier '>n' refers to the pixel ratio (%) whose error is larger than n, 'MAE' refers to the mean absolute error, and 'RMS' refers to the root mean squared error. The errors are averaged over all test frames of each datasets. '*' of each scores denotes the 2nd place.

Dataset	Omnidirectional stereo			Conventional stereo	
	Sunny [30]	Omni Things	Omni House	Scene Flow [16]	KITTI [7, 17]
MC-CNN [32]	✓				
SweepNet [30]	✓				
PSMNet [4]				✓	
PSMNet-ft				✓	✓
DispNet-CSS [12]				✓	
DispNet-CSS-ft				✓	✓
OmniMVS		✓			
OmniMVS-ft	✓	✓	✓		

Table 4: **Datasets used in each methods.** For experimental comparisons we use the published pre-trained weights for other methods (up of the dashed line). '-ft' denotes the fine-tuned versions.

House datasets for 16 epochs, with $\lambda = 0.003$ for 12 epochs and $\lambda = 0.0003$ for the rest. In our system with a Nvidia 1080ti, our OmniMVS takes 1.06s for processing which is quite fast, where MC-CNN [32] takes 1.97s, SweepNet [30] 6.16s, PSMNet [4] 1.79s, and DispNet-CSS [12] 0.57s.

5.2. Quantitative Evaluation

The error is measured by tie difference of inverse depth index as

$$E(\phi, \theta) = \frac{|\hat{n}(\phi, \theta) - n^*(\phi, \theta)|}{N} \times 100, \qquad (3)$$

which is the percent error of estimated inverse depth index from GT compared to all possible indices (N). We evaluate our approaches quantitatively on the available omnidirectional stereo datasets (Sunny, Cloudy, Sunset [30], OmniThings and OmniHouse).

We compare our method to the previous works of two types. The first type is spherical sweeping-based omnidirectional methods, and the second is stitching conventional stereo results into an omnidirectional one. We use the pre-trained weights of other methods in testing, and the training datasets for each method are described in Table 4.

Spherical sweeping ZNCC (zero-mean normalized cross correlation) and MC-CNN [32] compute the matching cost from 9×9 patches pair in the warped spherical images, and SweepNet [30] estimates the whole matching cost volume from the spherical images pair. Then, SGM [10] regularizes the cost volume with the smoothness penalties $P_1 = 0.1$ and $P_2 = 12.0$. As shown in Table 3, our end-to-end networks perform better in all datasets and metrics. Our OmniMVS builds more effective feature maps and learns better matching and regularization.

Stitching Conventional Stereo In order to estimate an omnidirectional depth, one can use a conventional stereo

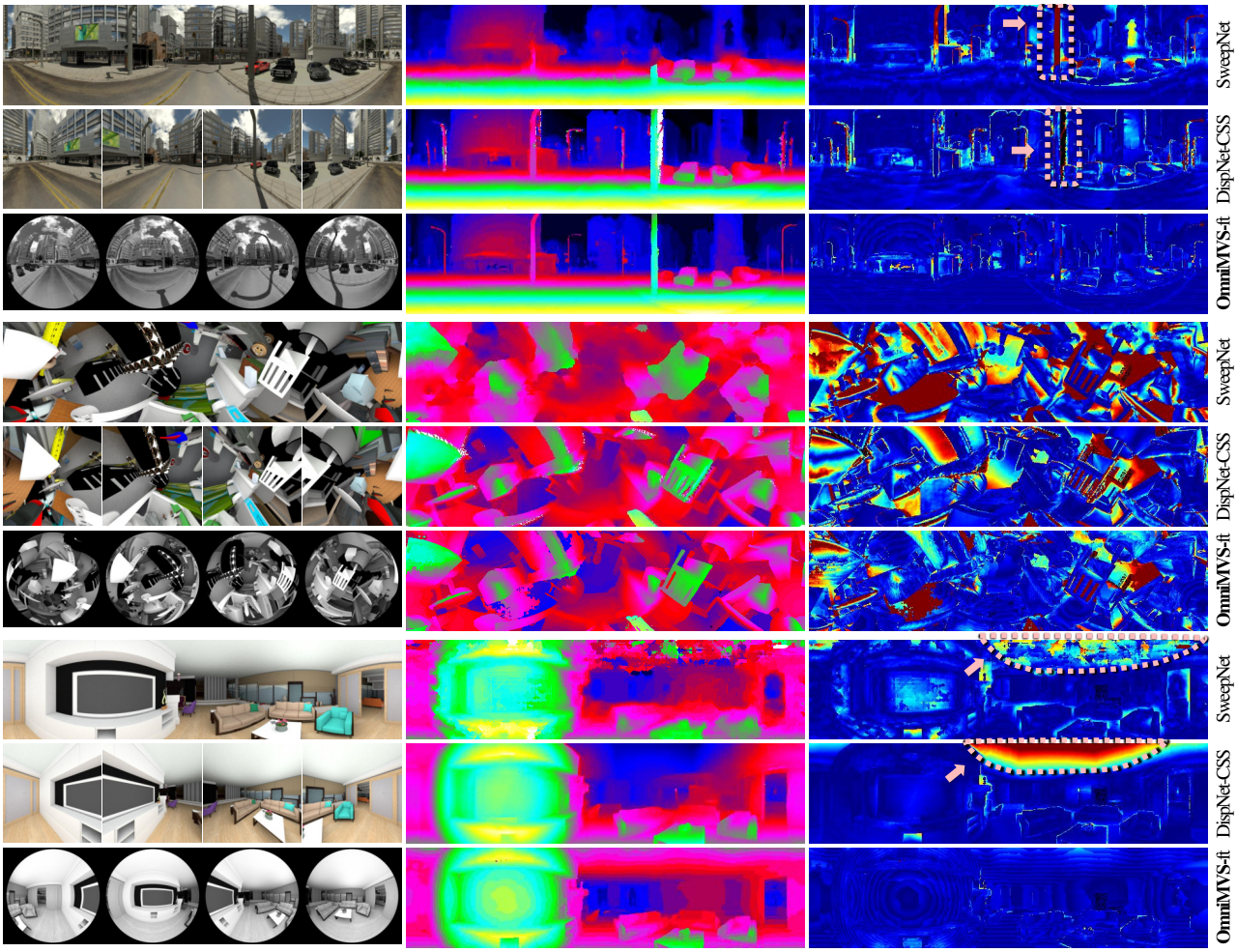


Figure 5: **Results on the synthetic data.** Left: reference panorama image, rectified left color images, and grayscale fisheye images. Middle: predicted inverse depth. Right: colored error map of inverse depth index error (blue is low and red is high).

method to compute disparities in different directions, and merge the depth maps into one panorama. As shown in Fig. 4, we generate four 120° rectified RGB image pairs from the fisheye images, and compute disparities by applying PSMNet[2] [4] or DispNet-CSS[3] [12]. Then all reconstructed 3D points are put in the rig coordinate system. For each pixel in the $H \times W$ spherical depth map, the closest 3D point which is projected within 1-pixel radius is chosen for output. The pixels without any points are ignored in the evaluation. As described in Table 4, we use the pretrained weights presented in their works. Table 3 shows that our networks achieved the best performance. Note that although OmniThngs and FlyingThings3D in SceneFlow [16] which share most of the objects, OmniMVS trained with OmniThings performs favorably to PSMNet or DispNet-CSS trained with SceneFlow.

[2]https://github.com/JiaRenChang/PSMNet
[3]https://github.com/lmb-freiburg/netdef_models

5.3. Qualitative Evaluation

Synthetic Dataset Figure 5 illustrates qualitative results of SweepNet [30], DispNet-CSS [12], and OmniMVS-ft on the synthetic datasets, Sunny, OmniThings and OmniHouse. As indicated by the orange arrows in Fig. 5, SweepNet with SGM [10] does not handle the multiple true matches properly (on the street lamp and background buildings) so the depth of thin objects is overridden by the background depth. Also they have difficulty in dealing with large textureless regions. Our network can successfully resolve these problems using global context information.

Real-world Data We show the capability of our proposed algorithm with real-world data [30]. In all experiments, we use the same configuration with the synthetic case and the identical networks without retraining. As shown in Fig. 6 and 7, our network generates clean and detailed reconstructions of large textureless or even reflective surfaces as well as small objects like people and chairs.

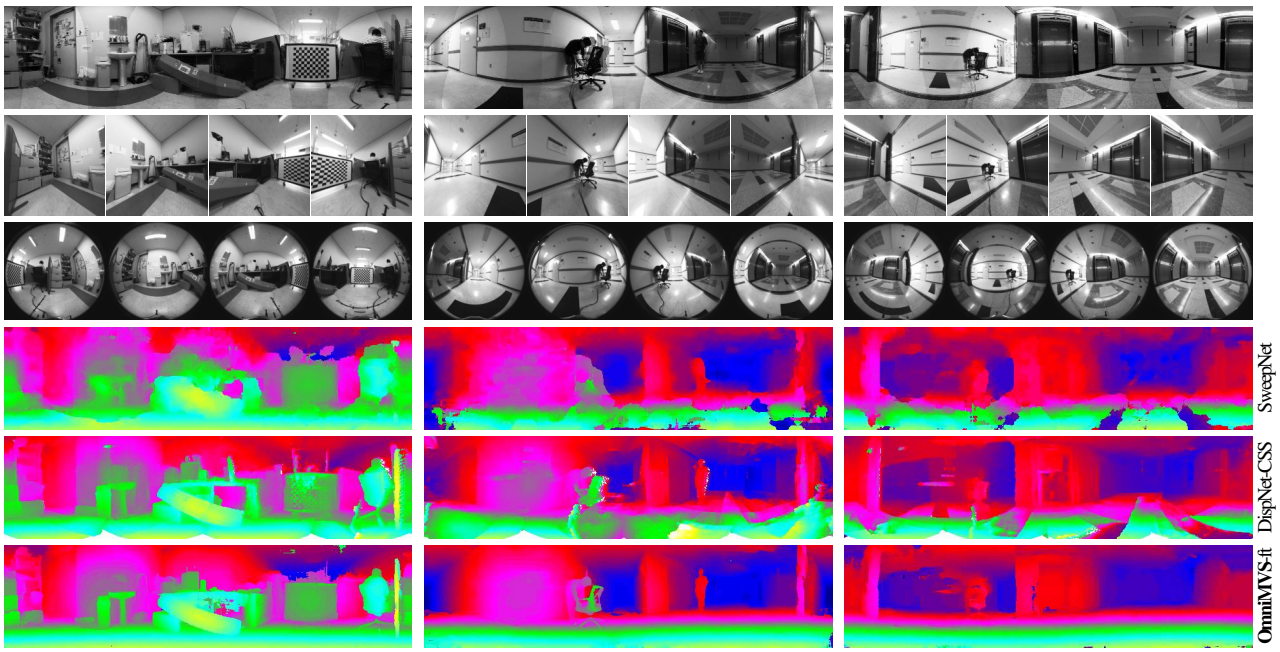


Figure 6: **Results on the real data.** From top: reference panorama image, rectified left images, input grayscale fisheye images, and inverse depth maps predicted by each methods. The reference panorama images are created by projecting the estimated 3D points from OmniMVS-ft to the input images.

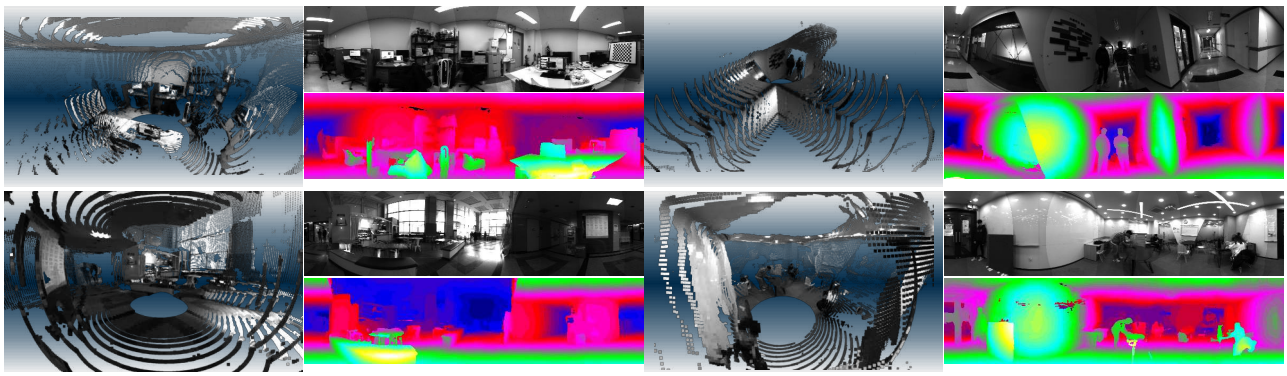


Figure 7: **Point cloud results.** Left: point cloud. Right: reference panorama image and predicted inverse depth estimated by the proposed OmniMVS-ft. Note that texureless walls are straight and small objects are reconstructed accurately. It also can handle generic rig poses (top-right).

6. Conclusions

In this paper we propose a novel end-to-end CNN architecture, OmniMVS for the omnidirectional depth estimation. The proposed network first converts the input fisheye images into the unary feature maps, and builds the 4D feature volume using the calibration and spherical sweeping. The 3D encoder-decoder block computes the matching cost volume, and the final depth estimate is computed by soft-argmin. Out network can learn the global context information and successfully reconstructs accurate omnidirectional depth estimates even for thin and small objects as well as large textureless surfaces. We also present large-scale synthetic datasets, Omnithings and OmniHouse. The extensive experiments show that our method outperforms existing omnidirectional methods and the state-of-the-art conventional stereo methods with stitching.

Acknowledgement

This research was supported by Next-Generation Information Computing Development program through National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT (NRF-2017M3C4A7069369), the NRF grant funded by the Korea government(MSIP)(NRF-2017R1A2B4011928), Research Fellow program funded by the Korea government (NRF-2017R1A6A3A11031193), and Samsung Research Funding & Incubation Center for Future Technology (SRFC-TC1603-05).

References

[1] Sameer Agarwal, Keir Mierle, and Others. Ceres solver. http://ceres-solver.org. 3

[2] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1, 2001. 2

[3] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 5

[4] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5410–5418, 2018. 1, 2, 6, 7

[5] Zhuoyuan Chen, Xun Sun, Liang Wang, Yinan Yu, and Chang Huang. A deep visual correspondence embedding model for stereo matching costs. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 972–980, 2015. 1

[6] Wenliang Gao and Shaojie Shen. Dual-fisheye omnidirectional stereo. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, pages 6715–6722. IEEE, 2017. 1, 2

[7] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3354–3361. IEEE, 2012. 2, 5, 6

[8] Fatma Guney and Andreas Geiger. Displets: Resolving stereo ambiguities using object knowledge. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4165–4175, 2015. 1, 2

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4

[10] Heiko Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on pattern analysis and machine intelligence*, 30(2):328–341, 2008. 1, 2, 3, 6, 7

[11] Heiko Hirschmuller and Daniel Scharstein. Evaluation of cost functions for stereo matching. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007. 2

[12] Eddy Ilg, Tonmoy Saikia, Margret Keuper, and Thomas Brox. Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 614–630, 2018. 1, 2, 6, 7

[13] Sunghoon Im, Hyowon Ha, François Rameau, Hae-Gon Jeon, Gyeongmin Choe, and In So Kweon. All-around depth from small motion with a spherical panoramic camera. In *European Conference on Computer Vision*, pages 156–172. Springer, 2016. 2

[14] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 66–75, 2017. 1, 2, 3, 4

[15] Wenjie Luo, Alexander G Schwing, and Raquel Urtasun. Efficient deep learning for stereo matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5695–5703, 2016. 1

[16] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4040–4048, 2016. 1, 2, 5, 6, 7

[17] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3061–3070, 2015. 2, 5, 6

[18] Jiahao Pang, Wenxiu Sun, Jimmy SJ Ren, Chengxi Yang, and Qiong Yan. Cascade residual learning: A two-stage convolutional neural network for stereo matching. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 887–895, 2017. 2

[19] Greire Payen de La Garanderie, Amir Atapour Abarghouei, and Toby P Breckon. Eliminating the blind spot: Adapting 3d object detection and monocular depth estimation to 360 panoramic imagery. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 789–807, 2018. 2

[20] Stephan R Richter, Zeeshan Hayder, and Vladlen Koltun. Playing for benchmarks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2213–2222, 2017. 2

[21] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3234–3243, 2016. 1

[22] Davide Scaramuzza, Agostino Martinelli, and Roland Siegwart. A flexible technique for accurate omnidirectional camera calibration and structure from motion. In *Computer Vision Systems, 2006 ICVS'06. IEEE International Conference on*, pages 45–45. IEEE, 2006. 3

[23] Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nešić, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In *German conference on pattern recognition*, pages 31–42. Springer, 2014. 2

[24] Daniel Scharstein and Richard Szeliski. High-accuracy stereo depth maps using structured light. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 1, pages I–I. IEEE, 2003. 2

[25] Miriam Schönbein and Andreas Geiger. Omnidirectional 3d reconstruction in augmented manhattan worlds. In *Intelli-*

gent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on, pages 716–723. IEEE, 2014. 1, 2

[26] Akihito Seki and Marc Pollefeys. Sgm-nets: Semi-global matching with neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 231–240, 2017. 1, 2

[27] Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2, 5

[28] Steffen Urban, Jens Leitloff, and Stefan Hinz. Improved wide-angle, fisheye and omnidirectional camera calibration. *ISPRS Journal of Photogrammetry and Remote Sensing*, 108:72–79, 2015. 3

[29] Yanchang Wang, Xiaojin Gong, Ying Lin, and Jilin Liu. Stereo calibration and rectification for omnidirectional multi-camera systems. *International Journal of Advanced Robotic Systems*, 9(4):143, 2012. 1, 2

[30] Changhee Won, Jongbin Ryu, and Jongwoo Lim. Sweepnet: Wide-baseline omnidirectional depth estimation. *arXiv preprint arXiv:1902.10904*, 2019. 1, 2, 3, 5, 6, 7

[31] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4353–4361, 2015. 2

[32] Jure Zbontar and Yann LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, 17(1-32):2, 2016. 1, 2, 3, 6

[33] Nikolaos Zioulis, Antonis Karakottas, Dimitrios Zarpalas, and Petros Daras. Omnidepth: Dense depth estimation for indoors spherical panoramas. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 448–465, 2018. 2